Phylogenetic Super-Networks from Partial Trees

Daniel H. Huson, Tobias Dezulian, Tobias Klöpper, and Mike A. Steel

Abstract—In practice, one is often faced with incomplete phylogenetic data, such as a collection of partial trees or partial splits. This paper poses the problem of inferring a phylogenetic super-network from such data and provides an efficient algorithm for doing so, called the Z-closure method. Additionally, the questions of assigning lengths to the edges of the network and how to restrict the "dimensionality" of the network are addressed. Applications to a set of five published partial gene trees relating different fungal species and to six published partial gene trees relating different grasses illustrate the usefulness of the method and an experimental study confirms its potential. The method is implemented as a plug-in for the program SplitsTree4.

Index Terms—Molecular evolution, phylogeny, partial trees, networks, closure operations.

1 Introduction

Traditionally, in molecular phylogenetics, 16S rRNA has been used as the phylogenetic marker of choice to infer the evolutionary history of a collection of different species [20] which we will refer to as the *species tree*.

A *gene* tree is generated by considering some specific gene that is present in all given species. Given the current and growing abundance of whole genome sequences (see, e.g., [13]), for an increasing number of species it is now possible to compute gene trees for many different genes. Comparison of individual gene trees with the corresponding putative species tree may be useful, e.g., when trying to determine whether a given gene may have been involved in *horizontal gene transfer*. Also, one may attempt to obtain a more reliable species tree by forming a consensus tree from a set of gene trees.

A more recent approach to the gene trees versus species tree problem is to compute a consensus network [8], [7], [10] that attempts to represent *all* phylogenetic signals present in the given set of gene trees, simultaneously, up to a given level of complexity. In the resulting network, regions of the evolutionary history that are undisputed within the set of gene trees appear tree-like, whereas regions containing conflicts are shown as "incompatibility boxes" whose "dimensionality" reflects the number of conflicting signals.

In practice, for a given set of taxa of interest, it is often the case that some of the genes under consideration are not present in all genomes or, although present, their sequence is unavailable. This problem is usually addressed by removing those taxa from the analysis for which one of the genes is missing.

Therefore, it would be desirable to have a method that takes as input a collection of *partial* trees defined on subsets of the full taxa set and produces as output a phylogenetic network representing all phylogenetic signals present in the input partial trees.

In this paper, we describe a first such *super-network method*, which we call the *Z-closure* construction, and demonstrate its utility both by an experimental study and also by application to biological data sets. In this approach, partial splits are repeatedly extended using the *Z-rule*, a simple binary modification rule, until a closure under this rule has been obtained. The number of computed splits is at most equal to the number of input splits and such a closure can be computed efficiently.

In Section 2, we briefly introduce the basic underlying concepts of splits and splits graphs. Then, in Section 3, we define the *complete* Z-closure and provide an efficient algorithm for computing a (fixed order) Z-closure. We then define two important properties and prove that one can be used to assign weights to the edges in the network and the other can often be used to compute a form of *strict consensus tree* embedded in the network in Section 4. This is followed, in Section 5, by a discussion of how to compute the weights of the network edges. The results of a first experimental study of the method are presented in Section 6. In Section 7, we apply the method to five different partial gene trees showing the phylogenetic relationships among different species of fungi, published in [14], [15]. In Section 8, we show an application to grasses, using partial gene trees published in [5]. This example exhibits the problem that the *dimensionality* of the resulting graph may be very high and we propose a solution to this problem in Section 9. Finally, we discuss a number of variants of our approach and some other possible application scenarios in Section 10.

We have implemented the Z-closure in the program SplitsTree4 [10], which is freely available from www-ab.informatik.uni-tuebingen.de.

- *D.H. Huson, T. Dezulian, and T. Klöpper are with the Center for Bioinformatics (ZBIT), Tübingen University, Sand 14, 72076 Tübingen, Germany.*
  E-mail: {huson, dezulian, kloepper}@informatik.uni-tuebingen.de.
- *M.A. Steel is with the Biomathematics Research Centre, University of Canterbury, Christchurch, New Zealand.*
  E-mail: m.steel@math.canterbury.ac.nz.

Manuscript received 30 Sept. 2004; revised 10 Dec. 2004; accepted 14 Dec. 2004.
For information on obtaining reprints of this article, please send e-mail to: tcbb@computer.org, and reference IEEECS Log Number TCBB-0161-0904.

2 Trees, Splits, and Networks

Suppose we are given a set of taxa $X = \{x_1, \ldots, x_n\}$. A (*phylogenetic*) *X-tree* $T = (V, E, \nu, \omega)$ is a connected, acyclic graph with node set V and edge set $E \subseteq 2^V$, together with a *node labeling* $\nu : X \to V$ and *edge weighting* $\omega : E \to \mathbb{R}^{\geq 0}$, such that every node of degree 1 receives a label. An X'-tree T is called a *partial* X-tree if $X' \subseteq X$ and we define $X(T) = X'$, that is, $X(T) \subseteq X$ is the set of all taxa that are mentioned in T.

Suppose we are given an X-tree T. Every edge $e \in E$ partitions the tree T into two components and, thus, defines a bipartitioning of X into two nonempty and disjoint sets A and B, with $A \cup B = X$. Any such bipartitioning of X is called an *X-split*, written as $S = \frac{A}{B}$ (or, equivalently, $S = \frac{B}{A}$) [1]. For an X-split S obtained from a tree in this way, we define the *weight* of S to be $\omega(S) = \omega(e)$. A split S is called *trivial* if $|A| = 1$ or $|B| = 1$.

An X'-split S is called a *partial* X-split if $X' \subseteq X$ and, sometimes, we will call S a *full* X-split if $X' = X$. For an X-split $S = \frac{A}{B}$ and taxon set $X' \subseteq X$, we define the split *induced* on X' as $S|_{X'} = \frac{A \cap X'}{B \cap X'}$. Note that this may yield an *improper* split $\frac{\emptyset}{X'}$. We say that a split $S = \frac{A}{B}$ is an extension of a second split $S' = \frac{A'}{B'}$ if $A' \subseteq A$ and $B' \subseteq B$, where at least one of the inclusions is proper.

Suppose we are given two taxa $x, y \in X$ and a set of X-splits Σ. We say that a split $S = \frac{A}{B}$ *separates* x and y if $x \in A$ and $y \in B$ or vice versa and we use $\Sigma(x, y)$ to denote the set of all splits $S \in \Sigma$ that separate x and y.

Let $\Sigma(T)$ denote the *split encoding of T*, that is, the set of all X-splits defined by edges in T. Two X-splits $S_1 = \frac{A_1}{B_1}$ and $S_2 = \frac{A_2}{B_2}$ are called *compatible* if one of the four following intersections is empty: $A_1 \cap A_2$, $A_1 \cap B_2$, $B_1 \cap A_2$, or $B_1 \cap B_2$—otherwise, they are said to be *incompatible*.

We have the following well-known result [2]: Suppose we are given an arbitrary set Σ of X-splits. Then, Σ is the split encoding of some X-tree T if and only if Σ is *compatible*, that is, if all pairs of splits in Σ are compatible.

The split encoding of trees plays an important computational role in phylogenetics. For example, given a set of X-trees $T_1, \ldots, T_k$, we obtain the split encoding of the *strict consensus tree* T_{strict} as the set of all X-splits that occurs in every input set $\Sigma(T_i)$. Similarly, we obtain the *majority consensus tree* $T_{majority}$ via the set of splits that occur in more than 50 percent of all input sets. Finally, we obtain a (*d-dimensional*) *consensus network* as the set of all splits that occur in a proportion of more than $\frac{1}{d+1}$ of all input sets [8].

Suppose we are given an arbitrary set Σ of X-splits, not necessarily compatible. *Every* such set of splits can be represented by a *splits graph* $G = (V, E, \nu, \omega, \sigma)$, which consists of a connected graph G with vertex set V and edge set $E \subseteq 2^V$, together with a node labeling ν, edge weighting ω, and a surjective edge coloring $\sigma : E \to \Sigma$. Additionally, we require that the coloring σ is *isometric*, that is, for each pair of nodes $v, w \in V$, every shortest path from v to w uses the same set $\Sigma(v, w) \subseteq \Sigma$ of edge colors and each such color is used precisely once. Moreover, we require that $\Sigma(\nu(x), \nu(y)) = \Sigma(x, y)$ for all pairs of taxa $x, y \in X$. Finally, we assume that $\omega(e) = \omega(\sigma(e))$ for all edges $e \in E$. For details, see [4].

Such a splits graph has the property that if one deletes all edges colored by a given split $S = \frac{A}{B} \in \Sigma$, then one obtains precisely two components, one containing $\nu(A)$ and the other containing $\nu(B)$. Thus, splits graphs contain *phylogenetic trees* as a special case and generalize them to a specific type of *phylogenetic network*. We have developed and implemented algorithms for constructing and visualizing splits graphs, see [4], [9], [10].

3 The Z-Closure Network

Suppose we are given a set of *partial* X-splits Σ. Our goal is to modify splits in Σ so as to obtain a collection of *full* X-splits. We propose to achieve this by repeatedly applying the following simple transformation, which we call the *Z-rule*:

> For any two splits $S_1 = \frac{A_1}{B_1} \in \Sigma$ and $S_2 = \frac{A_2}{B_2} \in \Sigma$:
> if $A_1 \cap A_2 \neq \emptyset$, $A_2 \cap B_1 \neq \emptyset$, $B_1 \cap B_2 \neq \emptyset$, and $A_1 \cap B_2 = \emptyset$,
> then
> replace S_1 and S_2 by $S_1' = \frac{A_1}{B_1 \cup B_2}$, and $S_2' = \frac{A_1 \cup A_2}{B_2}$.

In shorthand, we write $\frac{A_1}{B_1}\,\mathrm{Z}\,\frac{A_2}{B_2} \longrightarrow \frac{A_1}{B_1 \cup B_2}, \frac{A_1 \cup A_2}{B_2}$, where the three lines arranged in a "Z" connect those pairs of split parts that are required to have a nonempty intersection, hence, the name "Z"-rule. This rule was introduced by Meacham in the context of inferring phylogenies from multistate characters [3], [12], [17]. Note that application of the Z-rule will sometimes simply reproduce the two input splits S_1 and S_2, in which case, we say that the Z-rule *does not apply*.

We obtain *a (fixed order) Z-closure* $\bar{\Sigma}$ by repeatedly applying the Z-operation to all splits originally contained in, or derived from, the input set Σ, in some fixed order. We define the *complete Z-closure* as the set of all splits that occur in at least one (fixed order) Z-closure. To avoid excessive notation, we will also use $\bar{\Sigma}$ to denote the complete Z-closure, but will always distinguish sharply between *a* Z-closure and *the complete* Z-closure. In both cases, we are particularly interested in the set $\bar{\Sigma}^*$ consisting of all *full* X-splits contained in $\bar{\Sigma}$, together with all trivial X-splits, which we will also refer to as a or the Z-closure, respectively. We define a *Z-closure network* $Z(T_1, \ldots, T_k)$ for $T_1, \ldots, T_k$ to be a splits graph representing $\bar{\Sigma}^*$.

The following algorithm computes a (fixed order) Z-closure, maintaining all partial splits in an array `data` and using three sets of indices, `old`, `active`, and `new`, indicating which splits in the array were produced in an earlier, the previous, or the current iteration of the algorithm, respectively:

Algorithm 1 (Z-Closure)
Input: *A set of partial trees* $\mathcal{T} = \{T_1, \ldots, T_k\}$
Output: *A Z-closure* $\bar{\Sigma}^*$
Initialization:
    Let `data` *be an array initialized to the set of all nontrivial splits in* $\bigcup_i \Sigma(T_i)$.
    Let `old` *be a set of indices, initially empty*
    Let `active` *be a set of indices, initially empty*
    Let `new` *be a set of indices, initialized to the index set of* `data`
while `new` $\neq \emptyset$ do
    Append `active` *to* `old`, *set* `active` $=$ `new` *and*

set $\texttt{new} = \emptyset$
for each $i \in \texttt{old} \cup \texttt{active}$ do
for each $j \in$ do
Let $S_1 = \frac{A_1}{B_1} = \texttt{data}[i]$ *and* $S_2 = \frac{A_2}{B_2} = \texttt{data}[j]$
if $A_1 \cap A_2 \neq \emptyset$, $A_2 \cap B_1 \neq \emptyset$, $B_1 \cap B_2 \neq \emptyset$, *and* $A_1 \cap B_2 = \emptyset$ then
Define $S_1' = \frac{A_1}{B_1 \cup B_2}$ and $S_2' = \frac{A_1 \cup A_2}{B_2}$
if $S_1' \neq S_1$ then set $\texttt{data}[i] = S_1'$ and add i to new
if $S_2' \neq S_2$ then set $\texttt{data}[j] = S_2'$ and add j to new
Return the set of all X-splits in old ∪ active ∪ *all trivial X-splits.*

We claim:

Theorem 1. *Let $n = |X|$ and $m = |\Sigma|$. Algorithm 1 computes a Z-closure $\bar{\Sigma}^*$ in at most $O(nm^3)$ steps. The space requirement is $O(nm)$ and the resulting number of splits is at most $n + m$.*

Proof. The algorithm operates as follows: Originally, all splits are considered "new." In any iteration, the set of all splits deemed "new" in the previous iteration are considered "active" and are compared with themselves and with all "old" splits. The resulting "new" splits become the "active" splits of the next iteration, repeatedly, until no "new" splits are generated. In the worst case, each iteration of the algorithm will extend only one split by one taxon. There are m splits, each requires $O(n)$ iterations to extend to size n and each such iteration requires $O(m^2)$ comparisons, thus leading to a naive bound of $O(nm^3)$. As the algorithm operates "in place," the space requirement is simply the size of the input set, $O(nm)$. Moreover, the number of output splits is at most the number of input splits, m, plus the number of trivial splits on X, which is n. □

In practice, we can expect each split to be extended at least by one taxon in every iteration, leading to a runtime bound of $O(nm^2)$.

Algorithm 1 computes a (fixed order) Z-closure efficiently. It would be desirable to have an efficient algorithm for computing the complete Z-closure, but it is an open problem whether such an algorithm exists. The results described below indicate that a (fixed order) Z-closure is a very good approximation to the complete Z-closure. Moreover, any order-dependence of the algorithm can be addressed by running the Z-closure algorithm a number of times using random input orders and retaining all splits computed and our implementation of the method supports this feature. Moreover, the following result adds justification to the use of a (fixed order) Z-closure, see [3] for details:

Lemma 1. *If all input splits are compatible with each other, then any (fixed order) Z-closure is equal to the complete Z-closure.*

4 The Weak and Strong Induction Properties

Suppose we are given an input set of partial trees $\mathcal{T} = \{T_1, \ldots, T_k\}$. In the following, let X_i, Σ_i, and ω_i denote the taxa set, split encoding, and split weights for tree T_i, respectively, for all $i = 1, \ldots, k$.

We say that a split $S \in \bar{\Sigma}^*$ has the *weak* or *strong-induction property* if there *exists* a tree $T_i \in \mathcal{T}$ such that $S|_{X_i} \in \Sigma_i$ or if, for *every* tree $T_i \in \mathcal{T}$ such that $S|_{X_i}$ is a proper X_i-split, we have $S|_{X_i} \in \Sigma_i$, respectively. We say that $\bar{\Sigma}^*$ has the *weak* or *strong-induction property* if every split $S \in \bar{\Sigma}^*$ has the weak or strong-induction property, respectively.

The following result shows that the complete Z-closure does not contain any superfluous splits. Moreover, it is used in Section 5 to define the weights of the splits in $\bar{\Sigma}^*$:

Theorem 2. *The complete Z-closure of any set of partial X-trees $\mathcal{T} = \{T_1, \ldots, T_k\}$ has the weak induction property.*

Proof. We will show by induction that, for any split $S \in \bar{\Sigma}$, there exists an input tree T_i such that $S|_{X_i} \in \Sigma_i$. Let Σ^p denote the set of all splits obtained by p applications of the Z-rule (in some order). Induction start: Consider $S \in \Sigma^0 = \Sigma$. By definition of Σ, there exists a tree T_i with $S = S|_{X_i} \in \Sigma_i$. Induction step: Consider a split $S \in \Sigma^{p+1}$. If $S \in \Sigma^p$, then S has the desired property. If $S \in \Sigma^{p+1} \setminus \Sigma^p$, then S was obtained by extension of some split $S' \in \Sigma^p$ using the Z-rule. By the induction hypothesis, there exists a tree T_i with $S'|_{X_i} \in \Sigma_i$. As S is an extension of S', we have $S|_{X_i} = S'|_{X_i}$ and, thus, $S|_{X_i} \in \Sigma_i$. □

Suppose we are given a set of partial trees $\mathcal{T} = \{T_1, \ldots, T_k\}$. Let $\bar{\Sigma}^*_{SIP}$ be the set of all splits in $\bar{\Sigma}^*$ that have the strong induction property.

Often, although not always, $\bar{\Sigma}^*_{SIP}$ will be compatible and, thus, can provide a method for extracting a kind of *strict consensus tree* from the complete Z-closure $\bar{\Sigma}^*$.

5 Computing Weights

In Section 3, we described how to compute the set of splits $\bar{\Sigma}^*$ for the super-network $Z(T_1, \ldots, T_k)$. We now address the question of how to assign weights to the splits in $\bar{\Sigma}^*$ and, thus, to the edges of the network $Z(T_1, , T_k)$.

Suppose we are given a set of input trees $\mathcal{T} = \{T_1, \ldots, T_k\}$. Let X_i, Σ_i, and ω_i denote the taxa set, split encoding, and split weights for tree T_i, respectively, for all $i = 1, \ldots, k$.

Consider an X-split $S = \frac{A}{B} \in \bar{\Sigma}^*$. Let $I(S)$ denote the set of all $i \in \{1, \ldots, k\}$ such that the induced split $S|_{X_i}$ is contained in Σ_i. By Theorem 2, we have $I(S) \neq \emptyset$ and, so, we can assign a weight to S, based on the induced splits.

One proposal is to give S the smallest weight present, that is, to set $\omega(S) = \min\{\omega_i(S|_{X_i}) \mid i \in I(S)\}$. Or, similarly, one could use the mean weight, that is, to set $\omega(S) = \frac{1}{|I(S)|}\sum_{i \in I(S)} \omega_i(S|_{X_i})$. However, in both cases, we are assuming that the weights in the different input trees are all on the same scale, a requirement that often does not hold.

To address this problem, we propose using the *average relative length* of edges in the input set to assign weights to splits in the network. More precisely, we suggest defining

$$\omega(S) = \frac{1}{|I(S)|} \sum_{i \in I(S)} \frac{\omega_i(S|_{X_i})}{\bar{\omega}(T_i)},$$

where $\bar{\omega}(T_i)$ is the average weight of the edges in tree T_i. That is, each edge e in $Z(T_1, \ldots, T_k)$ is given the average value of $\frac{\omega_i(e)}{\bar{\omega}(T_i)}$ over all trees T_i that contain a "restriction" of e. Note that $\frac{\omega_i(e)}{\bar{\omega}(T_i)}$ is the ratio of the length of e in T_i to the average length of the edges in T_i.

Alternatively, let us say that an input set $\mathcal{T}$ has the *all pairs* property if, for every pair $x, y \in X$, there exists an

input tree $T_i \in \mathcal{T}$ that contains both x and y. If this property holds, then we can define a pairwise distance between any two taxa $x, y \in X$, e.g., as their minimum, mean, or median distance in the input set. We then can obtain split weights for $\bar{\Sigma}^*$ using a least squares fit of the distance matrix [19]. Again, this method uses distances from different trees and, so, may be problematic if the trees are scaled differently.

6 Experimental Study

The performance of a phylogenetic inference method is sometimes evaluated in a simulation study, see, e.g., [11]. This involves repeating the following three steps a sufficiently large number of times: First, input data is generated according to some model of evolution, guided by a specific *model* phylogeny. Second, this data is fed to the phylogenetic method as input. Third, the resulting tree is compared with the model tree to assess the accuracy of the method.

In this section, we report on the results of a simulation study that we have performed. Our experiments are run as follows: Suppose we are given a set of taxa $X = \{x_1, x_2, \ldots, x_n\}$. We first choose a single model tree M, which we call the *model species tree*. From this species tree, we obtain a set of h different *model gene trees* $M_1, \ldots, M_h$ by performing a number of SPR operations on each (Subtree Pruning and Regrafting as defined in, e.g., [18]), with the goal of producing model gene phylogenies that are related to, but different from, the model species tree.

To obtain a collection of partial trees, for each model gene tree M_i, we randomly select a subset of $X_i \subseteq X$ and let T_i denote the resulting induced *model partial gene tree*.

We then compute a Z-closure $\bar{\Sigma}^*$ for the set of all such partial trees $\mathcal{T} = \{T_1, \ldots, T_h\}$.

The accuracy of $\bar{\Sigma}^*$ is evaluated as follows: Any split $S \in \bar{\Sigma}^*$ for which no restriction is present in any input tree T_i could be considered a *false positive* partial split. However, by Theorem 2, the number of such false positives is always zero.

As the Z-closure is not a phylogenetic inference method but rather a method for summarizing a collection of partial trees within a consistent super-network, the main question is how successful is the method at representing the partial splits in the input set. A *false negative* is any split $S \in \Sigma(T_i)$ that is not *represented* in the Z-closure, that is, for which no split $S' \in \bar{\Sigma}^*$ exists with $S'|_{X_i} = S$.

The false negative rate will depend primarily on how large the partial trees are and how well they overlap, as well as how similar the trees are. In our simulations, we apply a varied number of SPR operations, between 0.1 and 3.2 on average, and, thus, obtain gene trees of varying degrees of similarity. Additionally, we assume that the partial trees cover a large proportion of the taxon set X and we choose the average size of the partial taxon sets to lie in the range 40-95 percent, with a standard deviation of 10 percent. Our study considers 10 gene trees. These choices are motivated by the following use-case: A phylogeny of a group of species is to be studied based on a small number of important genes. In practice, it is usually the case that some gene sequences are missing for some of the taxa.

The results of our study are summarized in Fig. 1. As expected, the simulations confirm that the number of false negatives depends strongly on the average coverage of the partial trees and also on the similarity of the underlying gene trees, represented here by the number of SPR moves performed. Surprisingly, when the average coverage is larger than 50 percent, then the performance appears to be practically independent of the level of similarity. Moreover, the rate of false negatives drops well below 10 percent. In summary, these results suggest that the method should work well on data sets with an average coverage of more than 50 percent, say, regardless of how similar the input trees are.

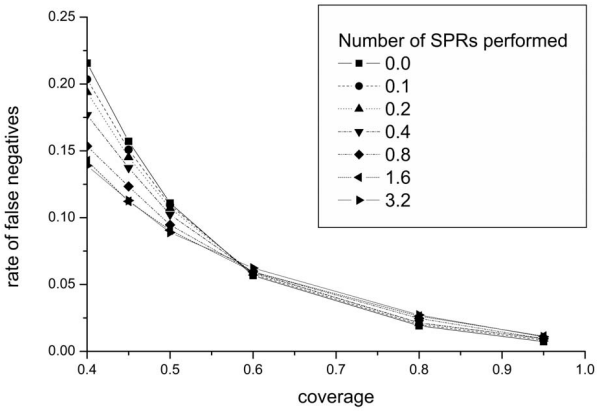


Fig. 1. Here, we plot the proportion of nontrivial partial splits present in the input that are not represented in the Z-closure-network as a function of the average proportion of taxa contained in the input trees and as a function of the number of SPR moves applied in the modification of the model gene trees. Every data-point represents the average score for 1,000 repetitions, each using a different 60-taxon model tree, randomly generated under the Yule-Harding model.

7 Application to Fungal Data

To illustrate the application of this method, we obtained five gene trees relating different fungal species from TreeBASE [16] that were published in [14], [15]. The three trees obtained from the first paper are based on the nuclear *internal transcribed spacer* (ITS), on the mitochondrial small subunit (SSU) ribosomal DNA (rDNA), and on a portion of the glyceraldehyde-3-phosphate gene (gpd), and are shown in Figs. 2a, 2b, and 2c, respectively. The two trees taken from the second paper are also based on the ITS and SSU sequences and are shown in Figs. 2d and 2e. Unfortunately, edge lengths were not available for the trees. In our experience, edge lengths greatly enhance the readability of the resulting network.

Calculation of a Z-closure network of the five trees took less than 20 seconds. The resulting graph is shown in Fig. 3. We also considered a sixth tree based on 18S rDNA that contains a large number of taxa not present in the other five. As to be expected, many of these taxa remained unresolved in the resulting network (not shown here).

The network depicted in Fig. 3 is based on a (fixed order) Z-closure. To determine how order-dependent the resulting network is, we reran Algorithm 1 a total of 1,000 times, using different random input orders. In Fig. 4, we see clearly that the input order has very little effect on the computed network. Indeed, in every single case, the derived Z-closure contains at least 97 percent of the union

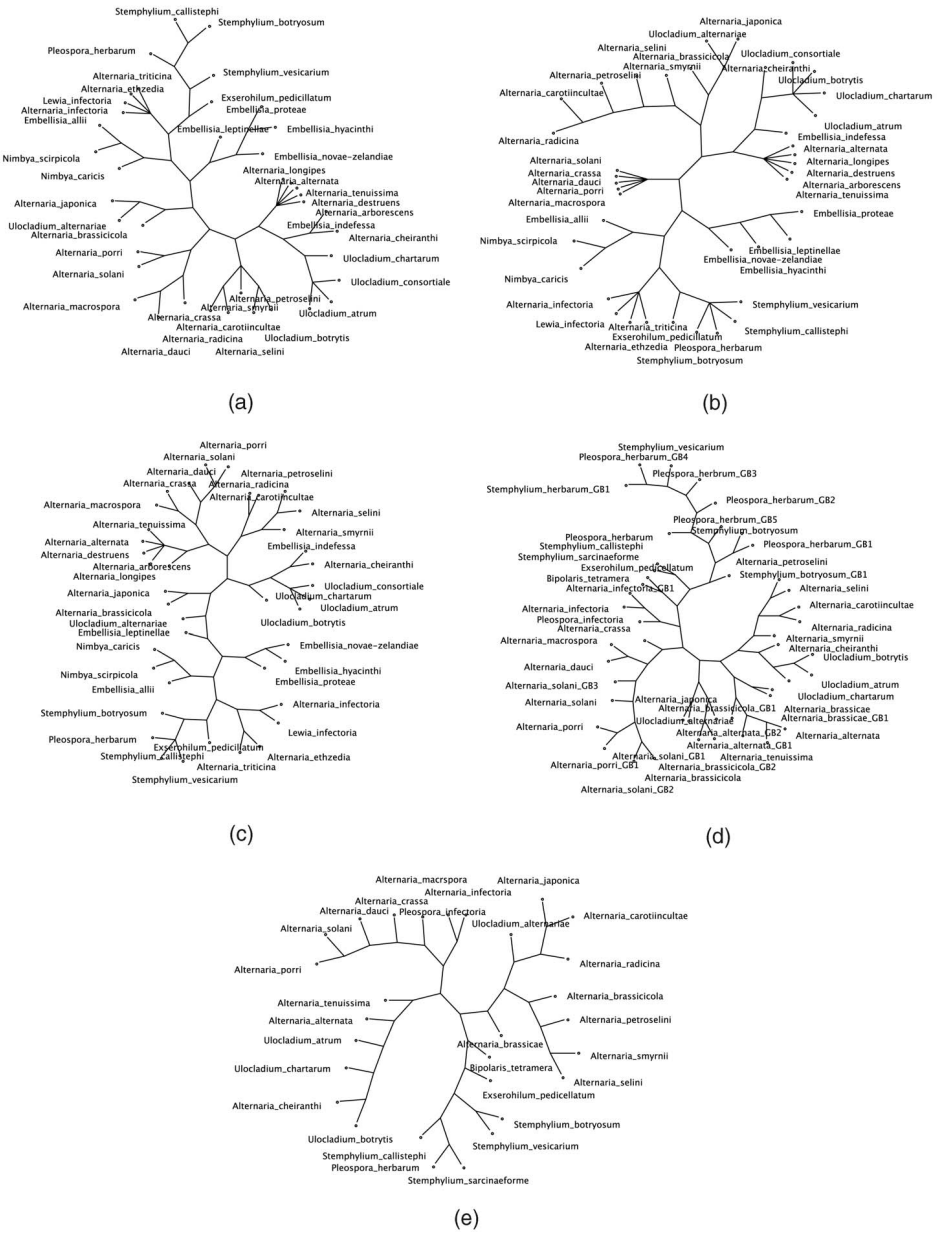


Fig. 2. The gene trees shown here are based on (a) ITS, (b) SSU, and (c) *gpd* sequences, as published in [14], and on (d) ITS and (e) SSU sequences, as published in [15]. These trees have varying numbers of taxa.

of all nontrivial splits obtained within the 1,000 runs. In this example, rerunning the algorithm a small number of times suffices to produce all 71 splits. Our naive implementation of the Z-closure algorithm took 25 minutes on a 1.2 GHz laptop to complete all 1,000 runs.

8 Application to Grasses

To provide a second illustration of this method, we obtained six published gene trees relating 66 different types of grasses (Poaceae) [5], covering 0.6 percent of all known species and 8 percent of the different genuses. The trees range in size from 19 to 65 taxa and are based on the following genes: *ndhF*, *rbcL*, *rpoC2*, *phyB*, and *GBSSI*. The authors of [5] generated the trees using a parsimony analysis and each tree is the strict consensus of between 1 and 33 most parsimonious trees.

In Fig. 5a, we show a splits graph representing all splits obtained from the six trees using the Z-closure method. Unfortunately, no branch lengths were available for the input trees. This example illustrates a potential problem with super-network methods. There may be many ways to extend a partial split to make it a full split and this can lead to a confusing graph, especially in the absence of branch lengths. To address this problem, we propose postprocessing the set of splits as described in the following section.

9 Dimension Filter

Let Σ be a set of X-splits. The *incompatibility graph* $IG(\Sigma) = (V, E)$ associated with Σ is obtained by setting $V = \Sigma$ and

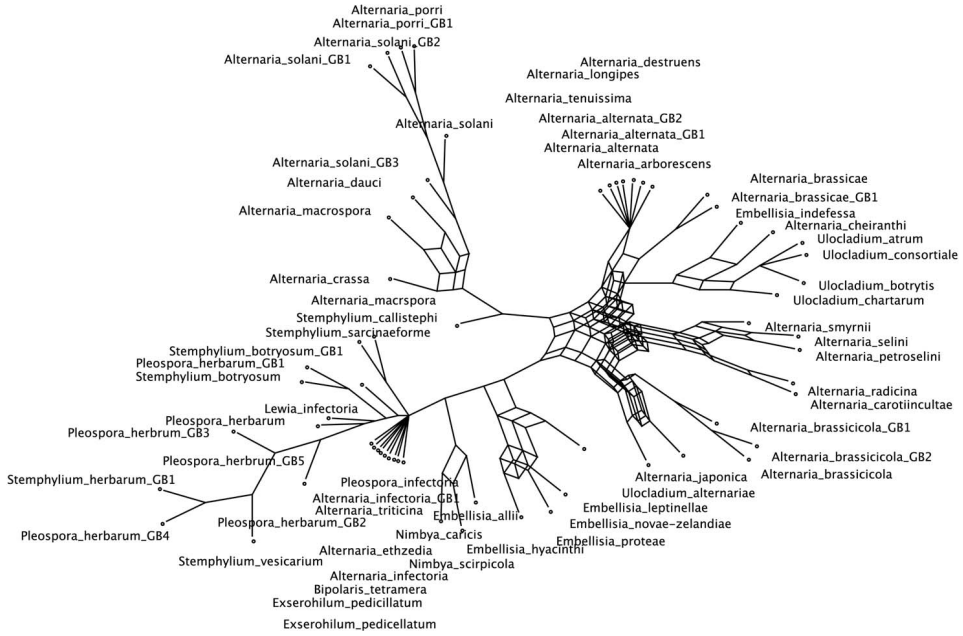


Fig. 3. A phylogenetic super-network on 63 taxa, obtained by applying the Z-closure method to the five partial trees depicted in the previous figure. This graph clearly shows on which parts of the phylogeny all partial gene trees agree and where there exist contradicting signals. Note that this type of graph is *not* a model of evolution, but rather a graphical summary of multiple phylogenies.

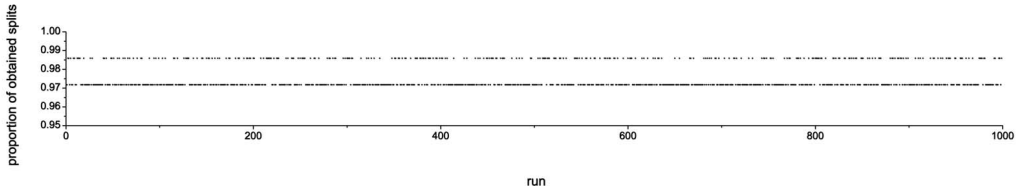


Fig. 4. For each of 1,000 different random input orders, we plot the number of splits obtained by the Z-closure algorithm as a proportion of the union of all splits obtained in all 1,000 runs. This is based on the input trees depicted in Fig. 2.

joining two nodes $S, S' \in V$ by an edge $\{S, S'\} \in E$ if and only if S and S' are incompatible.

To limit the complexity of the splits graph, for a given *dimension* d, we propose using a *dimension filter* that deletes a minimum number of nodes in the incompatibility graph such that the remaining graph contains no d-clique and, thus, that the corresponding splits graph contains no "d-dimensional cube." More precisely, we propose deleting a set of nodes $Q \subseteq V$ such that all d-cliques are destroyed and Q is of maximum "incompatibility," where the *incompatibility of* Q is defined as the sum of weights of all splits $S \in \Sigma \setminus Q$ that are incompatible with at least one split in Q, minus the sum of weights of all splits in Q.

Let $d(G)$ denote the maximal subgraph of G in which all nodes are contained in a d-clique. For fixed d, the graph $d(G)$ can be computed in polynomial time. We propose using the following greedy heuristic to remove a set Q of maximum incompatibility:

Algorithm 2 (Dimension Filter)
Input: Set of X-splits Σ *and an integer* $d > 0$
Output: Subset $\Sigma_d \subseteq \Sigma$ *such that* $IG(\Sigma_d)$ *contains no d-clique*
Construct $G = IG(\Sigma)$
Set $G' = d(G)$
Set $\Sigma_d = \Sigma$
while G' *is nonempty do*
  Choose a node S *in* G' *that is of maximal incompatibility*
  Remove S *from* G'
  Remove S *from* Σ_d
  Set $G' = d(G')$

To measure how well the remaining split set Σ_d approximates the original set Σ, we report the total weight of Σ_d as a percentage of the total weight of the original split set Σ. In Fig. 5, we illustrate the effect of dimension filtering with $d = 4$ and $d = 3$. We have implemented dimension filtering as a standard part of our SplitsTree software [10].

10 Discussion

We believe that the concept of a super-network will prove to be very useful, especially as applied to partial gene trees, where there is reason to believe that the underlying trees are incongruent and, thus, should not be "forced" into any particular super-tree. The Z-closure approach provides a simple and efficient method for computing super-networks, with some nice mathematical properties.

The Z-closure method has many potential applications. For example, in a first application, it has provided useful insights into the evolution of close relatives of *Arabidopsis*

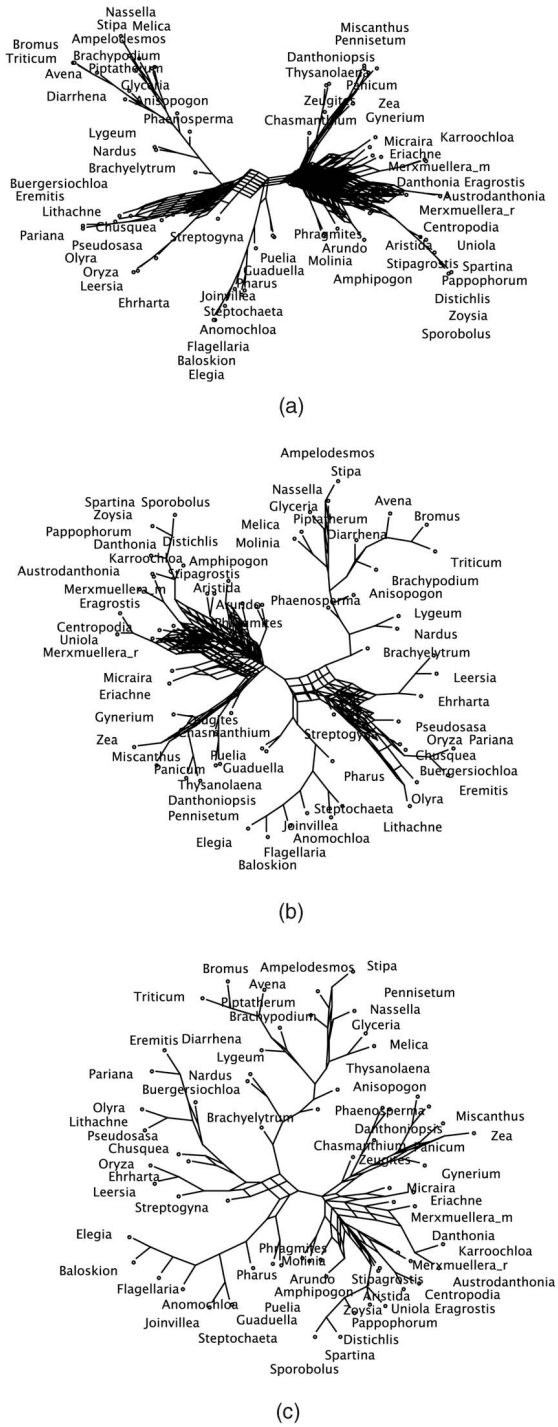


Fig. 5. In (a), we depict the full Z-closure of 171 splits computed for six gene trees published in [5]. In (b), we show 162 of the 171 splits (94.7 percent) obtained by using the dimension filter with $d = 4$. In (c), we show 143 of the 171 splits (83.6 percent) obtained by using the dimension filter with $d = 3$.

thaliana and these will be reported in a forthcoming paper (Lockhart et al., in preparation).

The approach formulated here can be extended in a number of different ways. First of all, note that the input can be an arbitrary collection of split systems and is not restricted to split systems coming from trees. In this case, Theorem 2 still holds.

Second, one can consider other rules of the type defined in [3], [12]. Our main motivation for focusing on the Z-rule is that this rule can be applied "in place" and does not increase the number of splits, leading to a particularly efficient algorithm. For example, another rule that one might additionally consider is the following:

> Suppose we are given $S_1 = \frac{A_1}{B_1}$ and $S_2 = \frac{A_2}{B_2}$ with $A_1 = A_2$ and $B_1 \cap B_2 \neq \emptyset$. Then, replace S_1 and S_2 by $S' = \frac{A_1}{B_1 \cup B_2}$.

In shorthand, $\frac{A_1}{B_1} = \frac{A_2}{B_2} \longrightarrow \frac{A_1}{B_1 \cup B_2}$. One draw-back of this *transitive* rule is that the condition $A_1 = A_2$ implies that the resulting set of splits will depend very strongly on the order of application, in contrast to the Z-rule.

One additional application of the Z-closure that we are currently studying is as the merge step in the "disk-covering method" [11]. A further potential application is to the problem of haplotype assignment: Given a set of n partial haplotypes, assign a complete haplotype to each [6]. Each partial haplotype is considered to be a partial split and complete haplotypes are obtained from the Z-closure network.

Acknowledgments

The authors would like to thank the New Zealand Institute for Mathematics and its Applications (NZIMA) for support under the *Phylogenetic Genomics* programme.

References

[1] H.-J. Bandelt and A.W.M. Dress, "A Canonical Decomposition Theory for Metrics on a Finite Set," *Advances in Math.*, vol. 92, pp. 47-105, 1992.

[2] P. Buneman, "The Recovery of Trees from Measures of Dissimilarity," *Math. Archaeological and Historical Sciences*, F.R. Hodson, D.G. Kendall, and P. Tautu, eds. pp. 387-395, Edinburgh Univ. Press, 1971.

[3] T. Dezulian and M. Steel, "Phylogenetic Closure Operations and Homoplasy-Free Evolution," *Proc. Meeting Int'l Federation of Classification Societies*, D. Banks, L. House, F.R. McMorris, P. Arabie, and W. Gaul, eds. pp. 395-416, 2004.

[4] A.W.M. Dress and D.H. Huson, "Constructing Splits Graphs," *IEEE/ACM Trans. Computational Biology and Bioinformatics*, vol. 1, no. 3, pp. 109-115, July-Sept. 2004.

[5] Grass Phylogeny Working Group "Phylogeny and Subfamilial Classification of the Grasses (poaceae)," *Annals Missouri Botanical Garden*, vol. 88, no. 3, pp. 373-457, 2001.

[6] E. Halperin and R.M. Karp, "Perfect Phylogeny and Haplotype Assignment," *Proc. Eighth Ann. Int'l Conf. Research in Computational Molecular Biology*, pp. 10-19, 2004.

[7] B. Holland, K. Huber, V. Moulton, and P.J. Lockhart, "Using Consensus Networks to Visualize Contradictory Evidence for Species Phylogeny," *Molecular Biology and Evolution*, vol. 21, pp. 1459-1461, 2004.

[8] B. Holland and V. Moulton, "Consensus Networks: A Method for Visualizing Incompatibilities in Collections of Trees," *Proc. Workshop Algorithms in Bioinformatics*, G. Benson and R. Page, eds., vol. 2812, pp. 165-176, 2003.

[9] D.H. Huson, "SplitsTree: A Program for Analyzing and Visualizing Evolutionary Data," *Bioinformatics*, vol. 14, no. 10, pp. 68-73, 1998.

[10] D.H. Huson and D. Bryant, "Estimating Phylogenetic Trees and Networks Using SplitsTree 4," manuscript in preparation, software available from www-ab.informatik.uni-tuebingen.de/software, 2004.

[11] D.H. Huson, S. Nettles, L. Parida, T. Warnow, and S. Yooseph, "The Disk-Covering Method for Tree Reconstruction," *Proc. Algorithms and Experiments*, pp. 62-75, 1998.

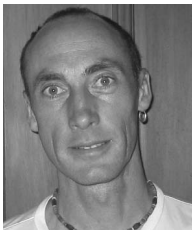
[12] C.A. Meacham, "Theoretical and Computational Considerations of the Compatibility of Qualitative Taxonomic Characters," *Numerical Taxonomy*, J. Felsenstein, ed., vol. G1, 1983.

[13] NCBI, "Microbial Complete Genomes Taxonomy," http://www.ncbi.nlm.nih.gov/PMGifs/Genomes/new_micr.html, 2003.
[14] B.M. Pryor and D.M. Bigelow, "Molecular Characterization of Embellisia and Nimbya Species and Their Relationship to Alternaria, Ulocladium, and Stemphylium," *Mycologia*, vol. 95, no. 6, pp. 1141-1154, 2003.
[15] B.M. Pryor and R.L. Gilbertson, "Phylogenetic Relationships among Alternaria and Related Fungi Based upon Analysis of Nuclear Internal Transcribed Sequences and Mitochondrial Small Subunit Ribosomal DNA Sequences," *Mycological Research*, vol. 104, no. 11, pp. 1312-1321, 2000.
[16] M.J. Sanderson, M.J. Donoghue, W. Piel, and T. Eriksson, "Treebase: A Prototype Database of Phylogenetic Analyses and An Interactive Tool for Browsing the Phylogeny of Life," *Am. J. Botany*, vol. 81, no. 6, p. 183, 1994.
[17] C. Semple and M.A. Steel, "Tree Recontruction via a Closure Operation on Partial Splits," *Computational Biology (Proc. JOBIM 2000)*, 2001.
[18] D.L. Swofford, G.J. Olsen, P.J. Waddell, and D.M. Hillis, "Phylogenetic Inference," *Molecular Systematics*, second ed., D.M. Hillis, C. Moritz, and B.K. Mable, eds., pp. 407-514, Sinauer Assoc., Inc., 1996.
[19] R.C. Winkworth, D. Bryant, P. Lockhart, D. Havell, and V. Moulton, "Biogeographic Interpretation Of Splits Graphs: Least Squares Optimization of Branch Lengths," *Systematic Biology*, 2004, submitted for publication.
[20] C.R. Woese, "Bacterial Evolution," *Microbiology Rev.*, vol. 51, pp. 221-272, 1987.

Daniel H. Huson studied mathematics at Bielefeld University and received the PhD degree in 1990. From 1990 to 1999, he held a variety of different research positions at Bielefeld University and was supported during this time by a two year DFG research scholarship. He then spent two years, 1997 to 1999, as a postdoctoral researcher working with Tandy Warnow at Princeton University. He then joined Celera Genomics as a senior research scientist working in Gene Myers group. Since 2002, he has been a professor of algorithms in bioinformatics at Tübingen University in Germany.

Tobias Dezulian studied computer science at Tübingen University. He received the diploma degree in 1999. After a three year excursion into business framework development, he is currently a PhD candidate in bioinformatics with Daniel H. Huson at Tübingen University.

Tobias H. Klöpper studied mathematics at the University of Göttingen. He received the diploma degree in 2003. He is currently a PhD student in bioinformatics with Daniel H. Huson at Tübingen University.



Mike A. Steel studied mathematics at Canterbury and Massey Universities (New Zealand) and received the PhD degree in 1989. From 1990-1993, he held various postdoctoral positions in Germany, the United States, and New Zealand and was appointed to a tenured position at University of Canterbury in 1994. He is currently a professor and director of the Biomathematics Research Centre at the University of Canterbury and is a principal investigator in the Allan Wilson Centre for Molecular Ecology and Evolution.